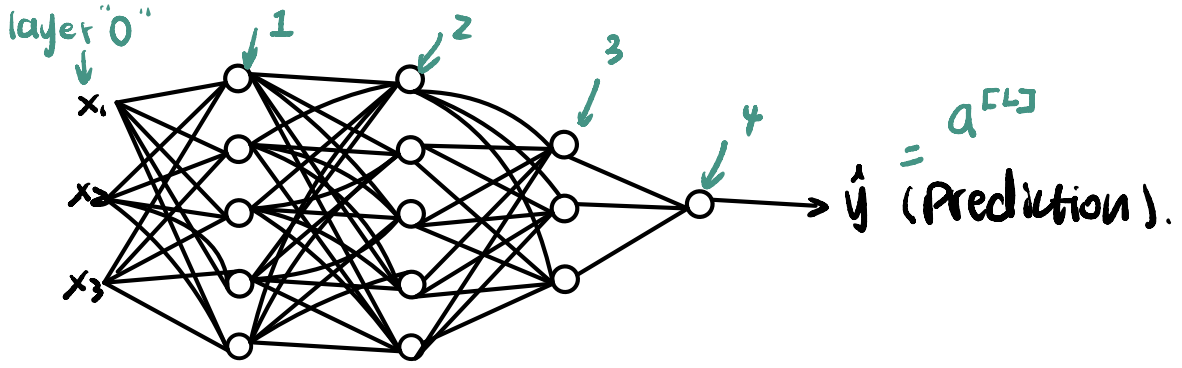
Deep L-layer Neural Network

Deep Neural Network notation:



$x = a^{[0]}$

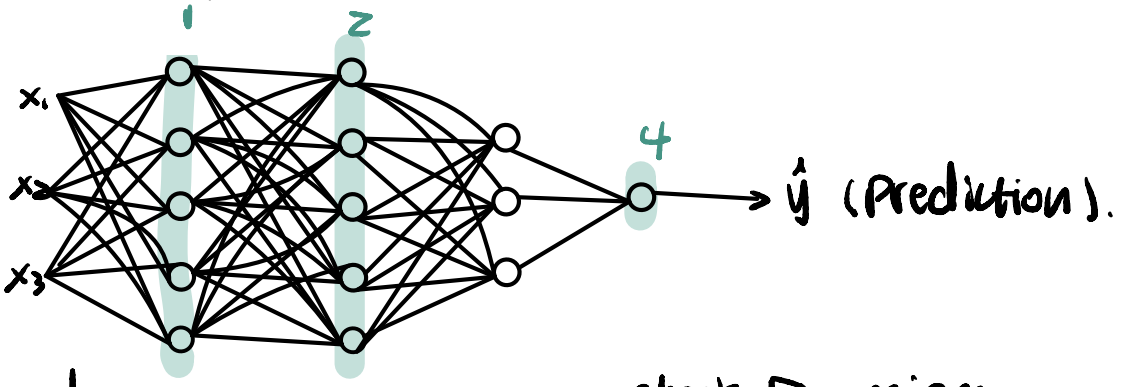
$l = 4$ (#layers), input layer's normally excluded from #layers

$n^{[l]}$ = # units in layer l

$a^{[l]}$ = activations in layer l

$a^{[l]} = g^{[l]}(z^{[l]})$, $w^{[l]}$, $b^{[l]}$ = weights for $z^{[l]}$.

Forward propagation:



vectorized:

$z^{[l]} = w^{[l]} A^{[l-1]} + b^{[l]}$

$A^{[l]} = g^{[l]}(z^{[l]})$.

Check Dimensions:

$w^{[l]} : (n^{[l]}, n^{[l-1]})$

$b^{[l]} : (n^{[l]}, 1)$

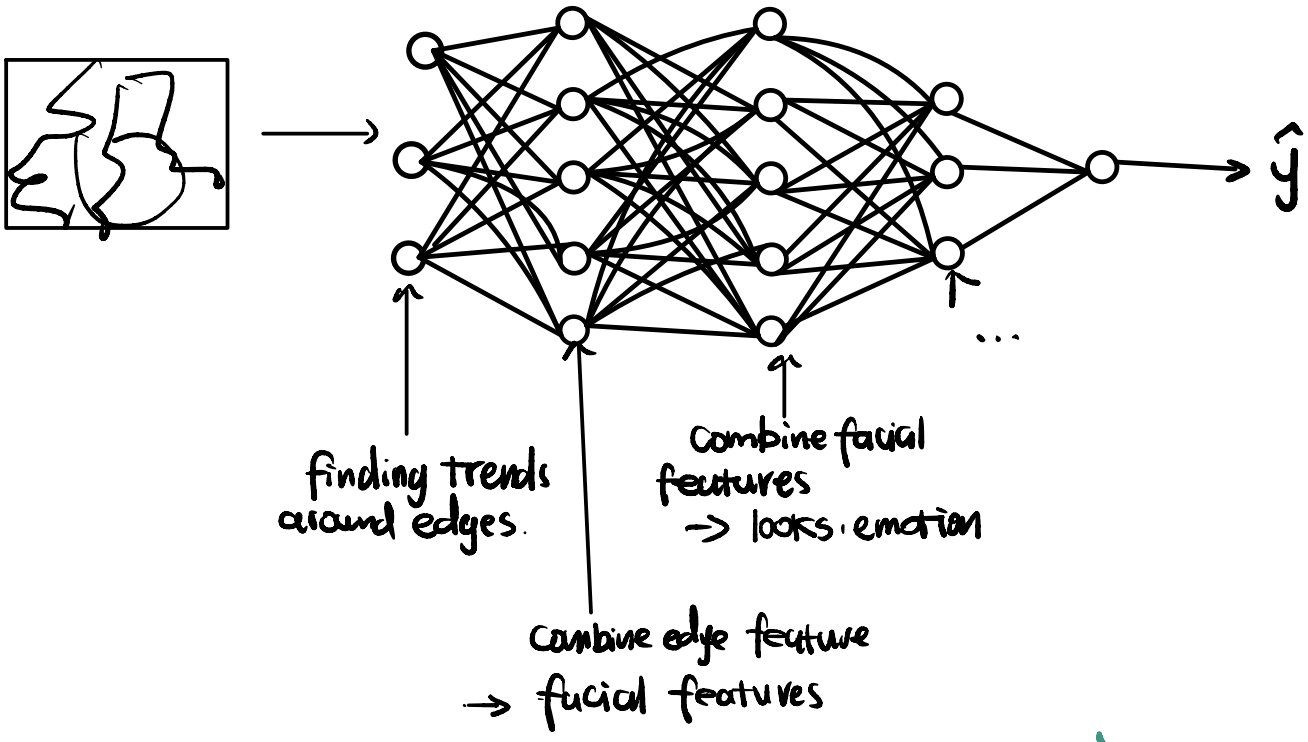
$dw^{[l]} : (n^{[l]}, n^{[l-1]})$

$db^{[l]} : (n^{[l]}, 1)$

$z^{[l]}, A^{[l]}, dz^{[l]}, da^{[l]} : (n^{[l]}, m)$

Intuition about deep representation.

eg. image recognition:



eg. Audio → Sound level Audio wave pattern. → Phonemes C A T .. → words → Sents Phrases.

Another Explaination: Circuit theory.

There are functions you can compute with a "small" L-layer deep neural network that shallower networks require exponentially more hidden units to complete.

hyperparameters:

params: $W^{[l]}$, $b^{[l]}$ ..

Hyperparams: learning rate α, #iterations, activation function, # hidden layers L, # hidden units $n^{[l]}$ ..

Building Blocks of deep neural network.

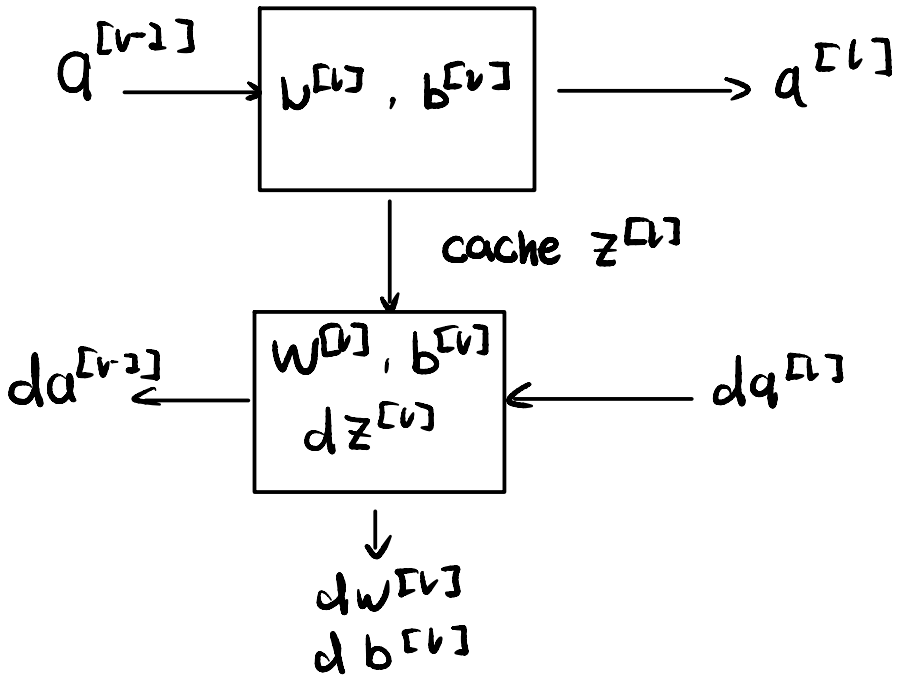
✳ Forward and backward functions

layer l: $W^{[l]}$, $b^{[l]}$

→ Forward: input $a^{[l-1]}$, output: $a^{[l]}$
cache: $z^{[l]}$

$$z^{[l]} = W^{[l]} a^{[l-1]} + b^{[l]}$$
$$a^{[l]} = g^{[l]}(z^{[l]})$$

→ Backward: input: $da^{[l]}$, output $da^{[l-1]}$
cache: $z^{[l]}$ $dW^{[l]}$
$db^{[l]}$



Backward propagation for layer l:

$$dz^{[l]} = dA^{[l]} * g^{[l]\prime}(z^{[l]})$$
$$dw^{[l]} = \frac{1}{m} dz^{[l]} A^{[l-1]T}$$
$$db^{[l]} = \frac{1}{m} np.sum(dz^{[l]}, axis=1, keepdims=True)$$
$$dA^{[l-1]} = W^{[l]T} \cdot dz^{[l]}$$